

In the Specification

Please amend paragraph the last paragraph beginning on page 3, line 29, as follows:

--Each of the addresses H_1, H_2, \dots, H_y is preferably obtained from the address A_0 by the following steps. Firstly, ~~we forming~~ a respective string S_n having the same number of bits as A_0 (according to present technology, 48) is formed. These S_n may just be respective sections of A_0 and in this case ~~we optionally select one S_n (say S_1)~~ is optionally selected and XORed [[it]] component-by-component with each of the other $y-1$ S_n , so that each of the other $y-1$ S_n is modified. Then each S_n (or modified S_n) is modulated with a respective set of Walsh codes (of the kind widely used in CDMA encoding for example). The y resultant strings are used in the same CRC which transformed A_0 to H_0 , to produce H_n . Due to the use of Walsh codes, the likelihood is higher of the H_n for different MAC addresses A_0 being different from each other.--

Please amend the second full paragraph on page 4, line 12, as follows:

--Specifically, this aspect of the invention may be expressed as ~~an switch~~ a switch including a memory for defining a look-up table having a plurality of addresses and a processor for associating MAC addresses with addresses of the look-up table, the processor being arranged to use each MAC address A_0 to generate $y+1$ look-up table addresses $H_0, H_1, H_2, \dots, H_y$ for y an integer greater than or equal to one, and according to at least one criterion to associate the address A_0 with a selected one of the addresses $H_0, H_1, H_2, \dots, H_y$.--

Please amend the last paragraph on page 5, line 18, as follows:

--Each of the 16-bit strings S_n is then used to generate a corresponding 48 bit string A_n , $n=1, \dots, 3$ by spreading/modulating the corresponding string S_n by using a respective code which is formed as a 16-bit concatenation of 3 different 16-bit Walsh codes. The nine 16-bit Walsh

codes are written $W_{n,m}$ $n=1,...,3$, $m=1,...,3$. The first three 3 components of A_n are formed by an XOR of the first component of S_n with the first three components of $W_{n,1}$ respectively. Similarly, the second three components of A_n are formed by an XOR of the second component of S_n with the second three components of $W_{n,1}$ respectively[. And]], and so on. The sixth three components of A_n are formed by XORing the sixth component of S_n by the last component of $W_{n,1}$ and the first two components of $W_{n,2}$ [. And]], and so on.--

Please amend the fourth full paragraph on page 7, line 18 as follows:

As a first example, although the algorithm has been shown trying just four look-up table addresses, this can be ~~generalised~~ generalized to y addresses (i.e. the algorithm above illustrates the special case of $y=3$). There are various ways in which the 48-bit MAC address A_0 can be used to generate y different addresses H_n , $n=1,..., y$ as will be clear to a skilled reader. The only way in which Figs. 2 and 3 need be varied in this case is that the test at steps 4 and 14 becomes whether n is less than y .--